CAR-TR-779
CS-TR-3499

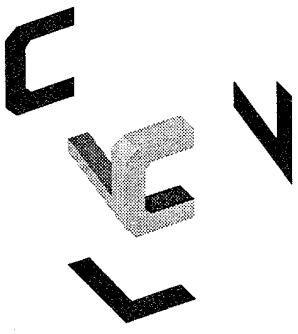
IRI-90-57934
N00014-93-1-0257
July 1995

3D Motion and Shape Representations in Visual Servo Control

Cornelia Fermüller
LoongFah Cheong
Yiannis Aloimonos

Computer Vision Laboratory
Center for Automation Research
University of Maryland
College Park, MD 20742-3275

COMPUTER VISION LABORATORY



CENTER FOR AUTOMATION RESEARCH

UNIVERSITY OF MARYLAND
COLLEGE PARK, MARYLAND
20742-3275

CAR-TR-779
CS-TR-3499

IRI-90-57934
N00014-93-1-0257
July 1995

3D Motion and Shape Representations in Visual Servo Control

Cornelia Fermüller
LoongFah Cheong
Yiannis Aloimonos

Computer Vision Laboratory
Center for Automation Research
University of Maryland
College Park, MD 20742-3275

Abstract

The study of visual navigation problems requires the integration of visual processes with motor control processes. Most essential in approaching this integration is the study of appropriate spatio-temporal representations which the system computes from the imagery and which serve as interfaces to all cognitive and motor activities. Since representations resulting from exact quantitative reconstruction have turned out to be very hard to obtain, we argue here for the necessity of of representations which can be computed easily, reliably and in real time and which recover only the information about the 3D world which is really needed in order to solve the navigational problems at hand. In this paper we introduce a number of such representations capturing aspects of 3D motion and scene structure which are used for the solution of navigational problems implemented in visual servo systems.

This research is supported by the National Science Foundation under Grant IRI-90-57934, the Office of Naval Research under Contract N00014-93-1-0257, and the Austrian "Fonds zur Förderung der wissenschaftlichen Forschung" Project No. S7003. A postgraduate scholarship from Tan Kah Khee Foundation is also gratefully acknowledged by LoongFah Cheong.

1 Introduction

Sensory motor activity, as it appears in nature, is largely controlled as a servomechanism. For instance, a person who is just learning to drive generally keeps the car on the road by fixing his attention on the edge of the road and comparing the location of this edge with some object on the car, such as the hood cap. If this distance is too small, the learner reacts by turning the steering wheel to the left;[1] if it gets too large, he reacts by turning the steering wheel to the right. It is characteristic of the learner that his driving consists of a series of oscillations about the desired position. As the driver improves, he introduces anticipation, or derivative control. In this condition a driver takes into consideration the rate at which he is approaching the correct distance from the edge of the road; eventually his control on the steering wheel becomes more sophisticated, possibly turning into a combination of proportional, derivative and integral control.

This and other examples from nature demonstrate that a continuous interplay between visual processing and motor activity is a characteristic of most existing systems that interact with their environments. In the psychological literature systematic investigations have been conducted on the role of vision in human motor control [41]. It has been concluded that for human motor coordination tasks, such as hand-eye coordination, there are two phases to a movement: an impulse phase that performs the initial motion (for instance, that moves the arm most of the way to the target) and a controlled phase that provides final adjustments. In the absence of vision, the accuracy of the movements is greatly reduced.

Similar phenomena, of course, arise in robotics. Most robotics systems of the past did not use perception, but were designed to perform a pre-specified set of motions. As the fields of Robotics and Computational Vision mature and increasingly complicated autonomous systems are studied, they have to deal with both perception and action. Computational theories are needed that explain perceptual capabilities, motor control procedures, as well as theories underlying the integration of vision and motor control into a working system.

Initial attempts at these difficult problems followed a modular approach. The goal of Computational Vision was defined as the reconstruction of an accurate description of the system's spatiotemporal environment. This description amounts to knowledge of the robot's 3D motion relative to any point in its environment and the depths or shapes of the surfaces in view. Assuming that this information can be acquired exactly, sensory feedback robotics was concerned with the planning and execution of the robot's activities. The problem with such separation of perception from action was that both computational goals turned out to be intractable. Complete visual reconstruction is generally an ill-posed problem, and can only lead to satisfying results if exact models of the robot's environment are available. On the other hand, spatial planning and motion control in 3D space are very sensitive to errors in the description of the spatiotemporal environment. Furthermore, this approach did not consider any complexity issues. Visual reconstruction is very time-consuming, the usual control problems require very expensive matrix computations, and many spatial planning problems are intractable. Real systems are bounded in their computational capacity and have to operate in real time.

It has been suggested in different fields—in Computer Vision, Artificial Intelligence, as

[1] If the car has a left-hand drive.

well as Robotics Engineering—that the major problem with this modular approach is that it does not enforce a tight coupling of perception and action. In Computer Vision the major approach of visual reconstruction, which aims for complete general representations, has been severely criticized. It has been argued that Computational Vision, in order to be successful, has to study vision for systems which are active and purposive [1, 2, 5, 6]. Recently a number of studies have been published which argue for a closer coupling by means of achieving solutions to a number of specialized visuo-motor control problems, usually problems for which a model of the environment is available [30, 31]. The field of AI brought forward the approaches of Connectionism and Behavior-Based Robotics [8], which propose to address the study of intelligent behavior through the construction of working mechanisms. In the classical engineering literature, we encounter the so-called approach of image-based control [4, 13, 34, 40]. The principle behind this approach is that instead of computing intermediate representations, directly available image measurements are used as feedback for the control loop. When referring to image measurements, image features are meant that supposedly can be easily extracted from the imagery, such as areas of faces, lengths of edges, angles, slopes of lines, or centroids of faces, if well-defined geometrical objects are considered. Most commonly, a number of feature points extracted from the image are tracked over time. The idea behind this is that the chosen image features can be directly related to the parameters of the robot's joints and thus a map, the so-called Perceptual Kinematic Map [20], is created which relates the joint space directly to the image space. However, image features of this kind, in general, are not easily extractable. The tracking of a number of points over a number of frames constitutes an ill-posed problem. Furthermore, even simple kinematic maps are no longer simple when it comes to inverting them. Thus the problems that could be managed in this way required very simple configurations, limiting the robot's degrees of mobility and considered only scenes for which geometrical models were available.

Here we argue that from the viewpoint of computational perception the essence of understanding the coupling of perception and action will come from understanding the appropriate spatiotemporal representations which the system computes from the imagery and which serve as interfaces to all cognitive and motor activities. These representations have to be easily obtainable, and they have to be computable reliably and in real time. The contribution of this paper lies in introducing such representations for 3D motion and shape. By considering three tasks requiring visual information of increasing complexity we introduce representations which capture a system's own motion, the motions of objects in the scene, and the shape of the scene. We also show how these representations can be integrated into visual servo systems based on dynamic feedback control.

The organization of this paper is as follows. Section 2 describes the visual reconstruction paradigm. It discusses the computations usually employed to derive from a sequence of images estimates of 3D motion and the shape of the scene in view. It then outlines the approach taken in this paper and briefly describes the three tasks addressed. The following sections are devoted to single tasks; first the task is described, second the representations computed from the visual information are given, and third the feedback control based on these representations is discussed. In particular Section 3 starts with a description of the robotic system used in all three tasks, followed by a description of the task of moving toward a fixed direction. In the subsections on the visual representations the global constraints

relating 2D flow field information to 3D egomotion information are described generally before it is explained how these constraints can be used in the servomechanism of a control system. Then the control is discussed. Section 4 outlines the second task, pursuing a moving target, which utilizes as visual representation the time to contact, a function of the speed of the robot and the relative depths of objects in the scene. Similarly, Section 5 is devoted to the third task, perimeter following, which uses as visual representation a function of the shape of the scene. Next, experiments related to all three tasks are described, and finally a summary of the work is given.

2 3D Motion and Shape Estimation

Problems of perception that require some 3D motion and shape information have usually been addressed in the context of general visual recovery. They were approached by first segmenting the image into areas of the same relative motion and then reconstructing the motion and shape of every point in the scene. The computational theory behind this approach, known as "structure from motion", suggests solving the problem in two stages. First, accurate image displacements between consecutive frames have to be computed, either in the form of point correspondences [15, 38] or as dense motion fields (optical flow fields) [3, 21, 23]. In a second step, the 3D motion and structure are derived from constraints due to the geometric transformation between the views relating the local 2D image measurements to 3D parameters [9, 22, 24, 26, 27, 35, 37]. Then, in order to solve the particular problems at hand, subsets of the computed structure and motion parameters have been utilized as inputs to non-visual cognitive processes, such as planning or control.

Despite the clear methodology and formalism of this computational theory, it turned out that in both computational steps extreme difficulties are inherently involved. The computation of optical flow and correspondence in the general case is an ill-posed problem and additional assumptions must be made in order to solve it. Recovering 3D motion from inexact or noisy flow fields has turned out to be a problem of extreme sensitivity. As a result, navigational problems have basically been treated as numerical analysis problems where sophisticated techniques (such as singular value decomposition, simulated annealing, Kalman filtering, maximum likelihood estimation, etc.) have been employed in order to estimate the 3D motion from the geometric and photometric constraints which relate local image motion to the 3D world.

To be successful in solving real-time navigational tasks we need to find a way of addressing the motion problem that uses easily obtainable information and ideally should use that information only to obtain the aspects of 3D motion and structure that we really need. For example, we may not need to know the exact motion parameters, but only approximations to the translation and the rotation, or whether there is significant motion at all, or whether there is significant rotation, or what the time to collision is, etc. Or we may be interested in using motion as a depth clue or shape clue; in that case, it is only the depth that matters, or only the derivatives of depth. Usually, qualitative information about depth is all we care about.

Since our goal is to study perception from a computational point of view, it is computational considerations that guide us in the study and development of visual representations.

Thus we classify visual representations according to the complexity of the computational models employed to relate image measurements to spatiotemporal representations and according to the complexity of the visual input used.

Obviously, not every motion is of the same complexity. Rigid motion is less complex than affine motion, and affine motion is less complex than a motion which can be modeled by a projective transformation, or than any other non-rigid motion.

It is less obvious, however, that not all rigid motion requires the same computational model and the same input. For example, whereas information about the egomotion of a system can be computed from global image information over the entire image plane, and thus image information can be employed in a redundant way, information about the motions of small objects in the scene can only be derived from a small field of view, and thus quantitative local measurements have to be used.

In this paper, by considering three visuo-servo motor control tasks, we present representations for 3D rigid motion and shape of increasing levels of complexity. In all three tasks we consider a robot system consisting of a body and a camera which can move independently of the body. The first and simplest task consists of changing the robot's direction of motion towards a fixed direction using visual information. For this task we need only partial egomotion information about the robot. This can be derived using the global constraints introduced in [18] and [19], which relate the sign of local image-flow information to the direction of translation and rotation. In the second task the robot must pursue a moving target while keeping a certain distance from the target. In addition to computing its own motion the system also has to derive some information about its speed relative to the target. The additional visual representation computed is the time to collision. In the third and last task, the robot has to follow a perimeter. This task requires the system to compute some form of depth information about the perimeter. The depth representation employed is less complex than the classical ones of (scaled) distance that are usually used. It is a function of scaled shape which can be derived without first computing 3D motion.

3 Task One: Moving Towards a Fixed Direction

The robotic system considered in all three tasks consists, as illustrated in Figure 1, of a body on wheels with a camera on top of the body. To describe the system's degrees of freedom we define two coordinate systems, attached to the camera and the robot. The robot moves on a surface and is constrained in its movement to a forward translation T_R (along z_R) and a rotation ω_R around the vertical axis, i.e., the y_R-axis. The camera is positioned along a vertical axis that passes through the center of rotation of the robot and it has two independent rotational degrees of freedom. Its orientation measured with respect to the coordinate frame of the robot is given by its tilt, θ_x, and its pan, θ_y; there is no roll, i.e. $\theta_z = 0$.

We first consider the simple task of moving towards a new direction. Referring to the mobile robot illustrated in Figure 2, the problem can be stated as follows: a robot, moving forward with speed S, is required to head towards a new direction, along which some feature $\mathbf{p}$ lies; $\mathbf{p}$ is selected beforehand by some higher level process. The robot first directs the

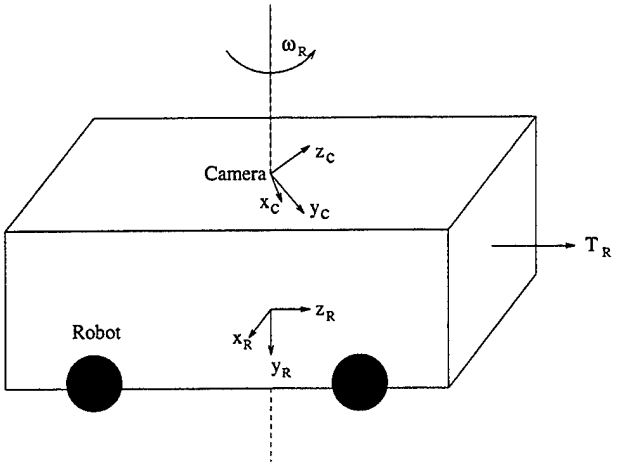


Figure 1: Camera and robot local coordinate systems.

camera at $\mathbf{p}$, so that the line of sight is now positioned along $\mathbf{p}$. We assume such gaze shifts are accomplished by fast saccadic movements.

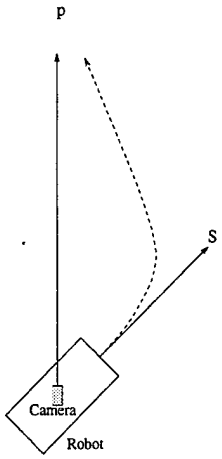


Figure 2: The robot, currently moving forward with speed S, aims to veer towards $\mathbf{p}$. The dotted path represents the trajectory generated in-flight by the servo system.

The robot must now make a series of steering decisions so that eventually its heading direction is aligned with where the camera is pointing. To be more accurate, since the robot moves on a surface and the feature in view can be at any height, we only want the direction of the forward motion and the direction of the heading to have the same projection on the xz-plane of the camera coordinate system, i.e., θ_y has to become zero. These steering movements are controlled by a servomechanism, which derives information from images captured by the camera.

The next section will be devoted to a discussion of the visual features, or more exactly, the visual patterns, employed by the servomechanism and the manner in which they are made use of. It will be followed by an analysis of the servomechanism itself.

3.1 Global motion patterns

The visual input that has been used to describe the computational analysis of visual motion is the optical flow field, which stems from the movement of light patterns in the image plane. Optical flow fields constitute a good approximation to the projection of the real 3D motion at scene points on the image [33, 39]. In general, however, accurate values of optical flow are not computable. On the basis of local information only the component of optical flow perpendicular to edges, the so-called normal flow, is well defined (the aperture problem; see Figure 3). In many cases it is possible to obtain additional flow information for areas (patches) in the image. Thus, the input that can be used by perceptual systems for further motion processing is some partial optical flow information.

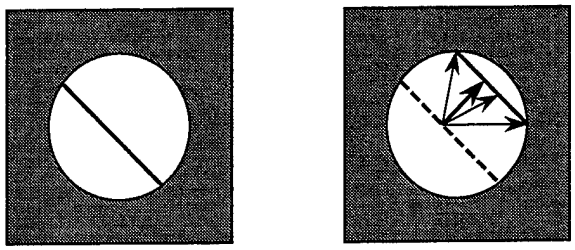


Figure 3: (a): Line feature observed through a small aperture at time t. (b): At time $t + \delta t$ the feature has moved to a new position. It is not possible to determine exactly where each point has moved to. From local measurements only the flow component perpendicular to the line feature can be computed.

The constraints that have been used in earlier work are mostly local ones. However, the utilization of image measurements from only small image regions is extremely error-prone: On the one hand, image measurements are very hard to compute accurately. Even if we just compute the normal flow, the projection of the retinal motion on the local image gradients, we need to use infinitesimal computations and have to approximate derivatives by difference quotients, and thus our computations can only be approximations. Much more difficult, however, is the computation of optical flow or disparity measurements, which requires us to employ some additional assumptions, usually smoothness assumptions, and thus we run into problems at motion and depth boundaries. On the other hand, even if we had reasonably accurate flow, it would not be the case that a small local change in flow implies a small change in three-dimensional motion. Completely different camera geometries produce locally similar disparity fields. For example, in an area near the vertical axis in the image plane, 3D rotation around the horizontal axis in the 3D world produces a flow field similar to the one produced by translation along the vertical axis in the 3D world.

Also, if we seek to provide solutions for real-time systems, we have to consider time constraints imposed on the actions the system performs. Computations that could not possibly be performed fast, such as linear, non-parallelizable algorithms that require a large number of steps, or optimization techniques involving a large number of iterations, are of little interest. In this category falls the computation of exact image displacements. Both the estimation of discrete disparities and the computation of optical flow require optimization techniques to

be invoked; and the more effort we put into deriving accurate measurements, by considering realistic situations dealing with motion boundaries and thus modeling discontinuities, the more computational steps we have to perform.

In [16, 19] it has been shown that 3D motion information can also be derived from a global structure in the motion field due to local image information much simpler than optical flow. New constraints defined on vectors in selected directions were found which manifest themselves as patterns in the image plane. The patterns are regions containing vectors of either positive or negative values, which are separated by conic sections and straight lines. In the following sections we will make use of some of these patterns, namely the "copoint patterns" and the "coaxis patterns". A short description of the corresponding constraints along with the basic equations relating 2D image motion to 3D motion and scene parameters is given below.

The 2D motion field on an imaging surface is the projection of the 3D motion field of the scene points moving relative to that surface. If this motion is rigid, it is composed of a translation $\mathbf{t} = (U, V, W)$ and a rotation $\omega = (\alpha, \beta, \gamma)$. We consider the case of a moving camera in a stationary environment with the coordinate system fixed to the nodal point of the camera and the image projection on a plane perpendicular to the Z-axis at distance f (focal length) from the center. Introducing the coordinates $(x_0, y_0) = (\frac{Uf}{W}, \frac{Vf}{W})$ for the direction of translation, the so-called FOE, we obtain the following well-known equations relating the velocity $\mathbf{u} = (u, v)$ of an image point to the 3D velocity and the depth Z of the corresponding scene point [27]:

$$\begin{aligned} u &= u_{\text{trans}} + u_{\text{rot}} = \\ &= (-x_0 + x)\frac{W}{Z} + \alpha\frac{xy}{f} - \beta(\frac{x^2}{f} + f) + \gamma y \qquad (1) \\ v &= v_{\text{trans}} + v_{\text{rot}} = \\ &= (-y_0 + y)\frac{W}{Z} + \alpha(\frac{y^2}{f} + f) - \beta\frac{xy}{f} - \gamma x \qquad (2) \end{aligned}$$

We consider the flow along certain directions, i.e. the the projection $\mathbf{u_n}$ of the flow $\mathbf{u}$ on direction $\mathbf{n} = (n_x, n_y)$ (unit vector), which is given as

$$\mathbf{u_n} = (\mathbf{u} \cdot \mathbf{n}) \cdot \mathbf{n}.$$

Thus we obtain u_n for the value of the vector $\mathbf{u_n}$ along the direction $\mathbf{n}$:

$$\begin{aligned} u_n &= \frac{W}{Z}((x - x_0)n_x + (y - y_0)n_y) \\ &\quad (\alpha\tfrac{xy}{f} - \beta(\tfrac{x^2}{f} + f) + \gamma y)n_x - (\alpha(\tfrac{y^2}{f} + f) - \beta\tfrac{xy}{f} - \gamma x)n_y. \end{aligned} \qquad (3)$$

By choosing particular directions, we define classes of vectors. In particular we consider here motion vectors in the direction of two classes of vectors, the "coaxis vectors" and the "copoint vectors".

The coaxis vectors defined with respect to a direction in space are described as follows: A line through the image formation center defined by the direction cosines (A, B, C) defines a family of cones with axis (A, B, C) and apex at the origin. The intersections of the cones

with the image plane give rise to a set of conic sections, called field lines of the axis (A, B, C), and the vectors perpendicular to the conic sections are are called the (A, B, C) coaxis vectors. Their direction is parallel to the vector (m_x, m_y), which is defined as

$$\begin{aligned}(m_x, m_y) &= ((-A(y^2+f^2)+Bxy+Cxf), \\ &\quad (Axy - B(x^2+f^2)+Cyf)).\end{aligned} \tag{4}$$

In order to establish conventions about the vector's orientation, a vector will be said to be of positive orientation if it is pointing in direction (m_x, m_y). Otherwise, if it is pointing in direction $(-m_x, -m_y)$, its orientation will be said to be negative (see Figure 4).

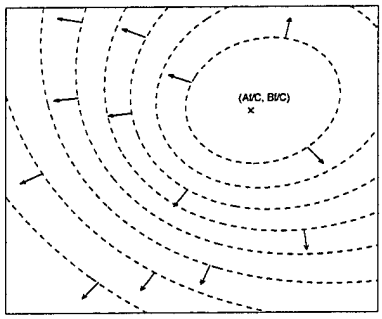


Figure 4: Field lines corresponding to an axis (A, B, C) and positive coaxis vectors (A, B, C).

If we consider for a class of (A, B, C) coaxis vectors the orientation of the translational components, we find that a second order curve $h(A, B, C, x_0, y_0; x, y) = 0$ (Figure 5a) separates the positive from the negative components, where

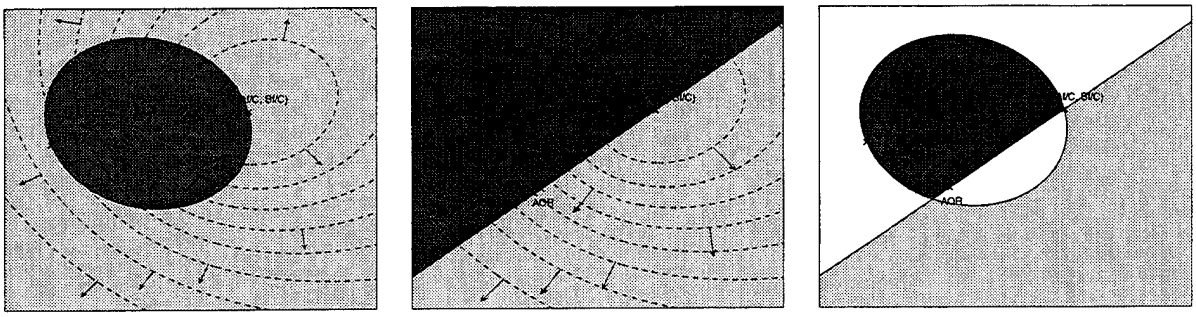
$$\begin{aligned}h(A, B, C, x_0, y_0; x, y) &= (x - x_0, y - y_0) \cdot (n_x, n_y) \\ &= x^2(Cf + By_0) + y^2(Cf + Ax_0) - xy(Ay_0 + Bx_0) \\ &\quad -xf(Af + Cx_0) - yf(Bf + Cy_0) + f^2(Ax_0 + By_0)\end{aligned} \tag{5}$$

Curve $h = 0$ passes through the FOE and is uniquely defined by the FOE's two image coordinates (x_0, y_0). Similarly, the positive and negative components of the (A, B, C) coaxis vectors due to rotation are separated by a straight line $g(A, B, C, \alpha, \beta, \gamma; x, y) = 0$, where

$$g(A, B, C, \alpha, \beta, \gamma; x, y) = y(\alpha C - \gamma A) - x(\beta C - \gamma B) + \beta A f - \alpha B f. \tag{6}$$

The line passes through the point where the rotation axis pierces the image plane (Figure 5b). The point, whose coordinates are $(\frac{\alpha f}{\gamma}, \frac{\beta f}{\gamma})$, is called the Axis of Rotation point (AOR). Combining the constraints due to translation and rotation, we obtain the following geometrical result: A second order curve separating the plane into positive and negative values and a line separating the plane into two half-planes of opposite sign intersect. This splits the plane into areas of only positive coaxis vectors, areas of only negative coaxis vectors, and areas in which the rotational and translational flow have opposite signs. In these last areas, no information is derivable without making depth assumptions (Figure 5c). The structure defined on the coaxis vectors is called the coaxis pattern.

For a second kind of classification, the copoint vectors, which are defined with respect to a point, similar patterns are obtained. The (r, s) copoint vectors are the normal motion



(a) (b) (c)

Figure 5: (a) The (A, B, C) coaxis vectors due to translation are negative if they lie within a second-order curve defined by the FOE, and are positive at all other locations. (b) The coaxis vectors due to rotation separate the image plane into a half-plane of positive values and a half-plane of negative values. (c) A general rigid motion defines an area of positive coaxis vectors and an area of negative coaxis vectors. The rest of the image plane is not considered.

vectors which are perpendicular to straight lines passing through the point (r, s) (see Figure 6). At point (x, y) an (r, s) copoint vector (o_x, o_y) of unit length in the positive direction is defined as

$$(o_x, o_y) = \frac{(-y + s, x - r)}{\sqrt{(x - r)^2 + (y - s)^2}}. \tag{7}$$

For the copoint vectors, the rotational components are separated by a second order curve into positive and negative values and the translational components are separated by a straight line. The structure defined on the copoint vectors is called the copoint pattern.

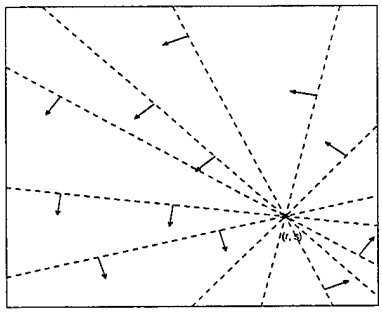


Figure 6: Positive copoint vectors (r, s).

Of particular interest for this application are the (r, s) copoint vectors for which the copoint (r, s) lies in infinity. The corresponding copoint vectors are all parallel to each other with gradient $\frac{n_x}{n_y} = -\frac{s}{r}$ (see Figure 7a,b). For these cases the line separating the translational components is perpendicular to the gradient vector (n_x, n_y) and has the following form:

$$k(n_x, n_y, x_0, y_0; x, y) = (x - x_0)n_x + (y - y_0)n_y \tag{8}$$

The curve separating the rotational components is a hyperbola given by

$$l(n_x, n_y, \alpha, \beta, \gamma; x, y) = (\alpha\frac{xy}{f} - \beta(\frac{x^2}{f} + f) + \gamma y)n_x + (\alpha(\frac{y^2}{f} + f) - \beta(\frac{y^2}{f} + f) - \gamma x)n_y \quad (9)$$

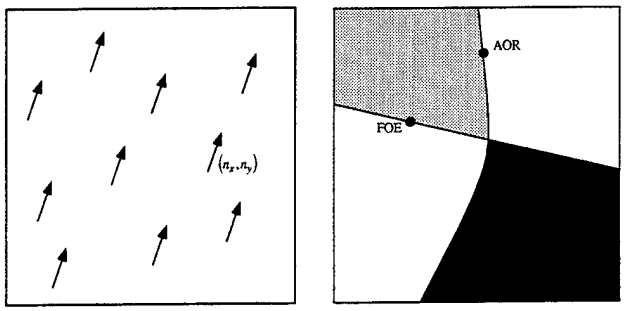


Figure 7: (a) Parallel copoint vectors. (b) Corresponding pattern.

The positions of the coaxis and copoint vectors in the image plane encode the parameters describing the axis of translation and the direction of the rotation axis. Thus, these constraints lead to formulating the problem of 3D motion estimation as a pattern recognition problem. If the system has the capability of estimating the the sign of the flow along the directions defined by various families of coaxis (or copoint) vectors, then by localizing a number of patterns, or more precisely, the boundaries of the regions separating positive and negative vectors, the system can find the axes of translation and rotation. The intersection of the coaxes' second order curves and the copoints' lines provides the FOE and the intersection of the coaxes' lines and the copoints' curves provides the AOR.

How much information will be available for pattern fitting, and thus how accurately the FOE and AOR can be localized, depends on the computability of flow information. If the system is able to derive optical flow then it is able to estimate the sign of the projection of flow along any direction, and thus for every pattern at every point information is available. If, however, the system is less powerful and can only compute the sign of the flow in one or a few directions, then patterns are matched as before. The difference is that information is not available for every point and consequently the uncertainty in pattern matching may be larger, and the FOE and AOR can only be located within bounds. In the simplest case, which requires the least amount of computational effort, the flow in only one direction, namely the one perpendicular to the local edge, is computed (the normal flow). But even this minimal amount of information can lead to rather small uncertainty in the motion estimation.

If the rigid motion estimation problem is considered for a passive system, a search in the appropriate parameter spaces has to be performed. Every single pattern is defined by three unknown parameters (a second order curve of two unknowns and a line of one unknown). A general rigid motion for which no information is available that could reduce the space of possible solutions thus requires a search in various 3D subspaces. If, however, the system is active, and if it has the capability to control its motor apparatus, the above described constraints may be utilized in much more efficient ways.

3.2 Using visual patterns in the servomechanism of a moving system

The retinal motion field perceived on the robot's camera is due to translation and rotation. The direction of translational motion is defined by the angle between the direction in which the robot is moving and the direction in which the camera is pointing. The rotation originates from body motion and is mainly due to the robot's turning around the y-axis. There could also be some rotation around the x-axis because the surface on which the robot is moving might be uneven, but there will be no or only very small rotation around the z-axis (cyclotorsion).

Recall that the goal of the visual task is to change the robot's motion such that the direction of the forward motion and the direction of the heading have the same projection on the xz-plane of the camera coordinate system. Stated in terms of motion parameters this means that we want the x-coordinate of the FOE to be zero, but we do not care about the y-coordinate.

Let us now investigate the patterns of of positive and negative flow vectors which correspond to such motion. We first consider the copoint patterns with parallel motion vectors. If the FOE is on the y-axis, i.e. $x_0 = 0$, (8) describing the translational vectors becomes

$$\begin{aligned} k(n_x, n_y, 0, y_0; x, y): \quad & x n_x + (y - y_0) n_y \quad = 0 \\ \text{or} \quad & y = -x \tfrac{n_x}{n_y} + y_0 \quad , \end{aligned} \tag{10}$$

which constitutes a line perpendicular to the gradient (n_x, n_y) with intercept y_0. In particular for the horizontal gradient direction ($n_x = 1$, $n_y = 0$) we obtain the simplified equation

$$k(1, 0, 0, y_0; x, y): \quad x \quad = 0$$

During the process of steering while the FOE is not aligned with the y-axis, the flow field due to translation is separated into positive and negative vectors through

$$k(n_x, n_y, x_0, y_0; x, y): \quad y \quad = -x \frac{n_x}{ny} + y_0 + x_0 \frac{n_x}{n_y}$$

and the horizontal flow vectors are separated by

$$k(0, 1, 0, y_0; x, y): \quad y \quad = y_0$$

The rotation around the y-axis is controlled by the robot's steering mechanism. Therefore the robot has knowledge about this rotation and can compensate for the resulting flow component perceived on the image by subtracting it from the visual motion field. Of course, we cannot assume the exact amount of rotation around the y-axis to be known, but we can assume that we know a good approximation to it. This additional knowledge makes one of the patterns, namely the copoint pattern with gradient $(1, 0)$, particularly suitable for fast estimation of the parameters to be controlled.

Let us consider the rotational components of the copoint pattern with gradient $(1, 0)$: Within the visual field of view the rotation around the x-axis gives rise to flow vectors which

are mostly parallel to the y-axis and thus perpendicular to the chosen gradients, As a result the components along the gradient direction $(1,0)$ are close to zero. The remaining (not derotated) rotation around the x-axis is nearly parallel to the gradients and thus causes a small, nearly constant component to be added to every flow vector. In summary, the contribution of the rotation to the pattern can be described as follows: The line in the translational pattern will be shifted by a small amount in the direction defined by the sign of the (not derotated) rotation around the x-axis.

For the purpose of the servoing task it will be sufficient to approximate the copoint pattern with gradient $(1,0)$ by its translational flow field components. Using this approximation gives us the advantage of deriving the x-component of the FOE with very little effort; we just fit a line perpendicular to the gradient direction separating positive from negative vectors. This approximation will not affect the successful accomplishment of the task. As the robot approaches its goal, the steering motion it has to apply becomes smaller and smaller and thus the additional rotational flow field component also decreases, which in turn allows the FOE to be estimated more accurately.

If it is certain that the rotation around the x-axis is also very small, then any other copoint pattern with some gradient (n_x, n_y) could be used in addition to estimate the FOE's coordinates using the approximation of considering the pattern to be translational.

Instead of utilizing copoint vectors we could equally well employ a class of coaxis vectors, namely those which correspond to axes in the XY-plane, the $(A,B,0)$ coaxis patterns. For these patterns the hyperbola separating the positive from the negative translational vectors becomes

$$h(A,B,0,x_0,y_0;x,y) : By_0x^2 + Ax_0y^2 - (Ay_0 + Bx_0)xy - Af^2x - Bf^2y + f^2Ax_0 + Bf^2y_0 = 0 \tag{11}$$

Since within the field of view f^2 is much larger than the quadratic terms in the image coordinates (x^2, y^2, xy), (11) can be approximated by $\hat{h}$ as

$$\begin{aligned} \hat{h}(A,B,0,x_0,y_0;x,y): \quad & f^2(-Ax - By + Ax_0 + By_0) = 0 \\ \text{or} \quad & y = \tfrac{A}{B}x_0 + y_0 - \tfrac{A}{B}x \end{aligned}$$

which describes a line with slope $\frac{-A}{B}$ and intercept $\frac{A}{B}x_0 + y_0$.

If $x_0 = 0$ the intercept is y_0. Again, one of these patterns, which allows to directly derive x_0, is of particular interest to us. This is the pattern corresponding to the axis $(1,0,0)$. We call this pattern the α-pattern and the corresponding vectors the α-vectors, since they do not contain any rotation around the x-axis (denoted in the equations by α).

$$\begin{aligned} h(1,0,0,x_0,y_0;x,y): \quad & x_0y^2 - y_0xy - xf^2 + x_0f^2 = 0 \\ \text{which simplifies to} \quad & x = x_0 \end{aligned}$$

The α-vectors do not contain any rotation around the x-axis and the components due to rotation around the y-axis are nearly constant. As in the previous case we can approximate the α-pattern by its translational component, which is of a particular simple form. And again, if we know that rotation around the x-axis is small, to obtain more data we can employ many $(A,B,0)$ coaxis patterns in the estimation of the FOE.

3.3 Servo system

For the purpose of our analysis, we will first consider our servomechanism to be a continuous-control linear system. The input signals encountered in the present application are step functions. Thus, a natural way to approach the design of our servomechanism is to consider the performance of the system in response to a step function input. Indeed, if our servomechanism is truly linear, its performance characteristics are completely summarized by its response to a step-function input.

Typical ways in which such an input is used include: proportional control, derivative control, integral control, proportional plus derivative control, and so on. The effect of the proportional gain, K_p, is to drive the system at higher velocities when the error is larger. The derivative gain, K_d, helps accelerate the response when it is falling behind and decelerates the response when it is overtaking the stimulus, which can speed up the response. However, a higher derivative gain produces an oscillatory response. The integral gain K_i, on the other hand, compensates for delays and disturbances. In the following case study, only a proportional controller is considered, though in practice a PID controller could be used to achieve better performance.

To set up the control loop equation, we must relate the robot's motion to the image motion. Referring to the coordinate systems defined in Figure 1, we denote the velocity of the robot's forward translation by S and the velocity of its rotation around the y-axis by β. θ_x and θ_y are the pan and tilt of the orientation of the camera with respect to the coordinate frame of the robot.

Using the subscripts R and C to denote the robot and the camera respectively, we can express the motion of the robot in the camera's coordinate frame as follows. First of all, let $\vec{P}$ be the position vector that relates the origins of the two coordinate frames. The rotation matrix ${}^{C}R_{R}$ that relates the orientations of the frames is of the following form:

$$\begin{pmatrix} \cos\theta_y & 0 & \sin\theta_y \\ -\sin\theta_y \sin\theta_x & \cos\theta_x & \cos\theta_y \sin\theta_x \\ -\sin\theta_y \cos\theta_x & -\sin\theta_x & \cos\theta_y \cos\theta_x \end{pmatrix}$$

Using T to denote translation, and ω to denote rotation, we can then express the motions of the robot and that of the camera in their respective coordinate frames as follows:

$$\begin{aligned} \vec{T_R} &= (0,0,S)^T \\ \vec{\omega_R} &= (0,\beta,0)^T \\ \vec{T_C} &= {}^{C}R_R(\vec{T_R} + \vec{\omega_R} \times \vec{P}) \\ &= (S\ \sin\theta_y, S\ \cos\theta_y \sin\theta_x, S\ \cos\theta_y \cos\theta_x)^T \\ \vec{\omega_C} &= {}^{C}R_R\ \vec{\omega_R} \\ &= (0, \beta\ \cos\theta_x, -\beta\ \sin\theta_x)^T \end{aligned}$$

Thus the coordinates of the FOE (x_0, y_0) that we computed for the camera motion are related to pan and tilt as follows:

$$x_0 = \frac{\tan\theta_y}{\cos\theta_x} f \tag{12}$$

$$y_0 = \tan\theta_x\ f \tag{13}$$

Our servomechanism should steer the robot in such a way that θ_y becomes zero. Referring to (13), it is noted that if the value of θ_x is known, we can compute the value of θ_y, thereby allowing the servo system to directly regulate it to zero. However, we do not have to assume knowledge of θ_x. It can be seen that regulating x_0 to zero would accomplish the same goal.

The position of x_0 is used as the input to the servo system to control the amount of steering the robot has to perform. If the servo system is operated with a proportional controller, the rotational speed, β, of the robot will be given by

$$\beta = Kx_0$$

Writing β as $\frac{d\theta_y}{dt}$, and substituting (13) for x_0, we obtain

$$\frac{d\theta_y}{dt} = \frac{\tan\theta_y}{\cos\theta_x} f \tag{14}$$

It is instructive to perform some simplifications so as to allow us to analyze how the aforementioned servo system responds to a step input. Since θ_x, the tilt, is usually very small, possibly slowly time-varying due to the camera's fixation on the feature **p**, we treat the denominator term $\cos\theta_x$ as a constant whose value is near 1. Furthermore, for the FOE to lie inside the image, θ_y must be smaller than half the field of view of the camera. Thus even for a camera with a wide field of view (on the order of 70°) $\tan\theta_y$ is well approximated by the linear term, θ_y. This approximation will become better as we approach the straight ahead direction, since θ_y tends to zero. With these two simplifications, we have $x_0 = K_v\theta_y$, where K_v is given by $\frac{f}{\cos\theta_x}$. The overall servo system can be represented schematically as in Figure 8.

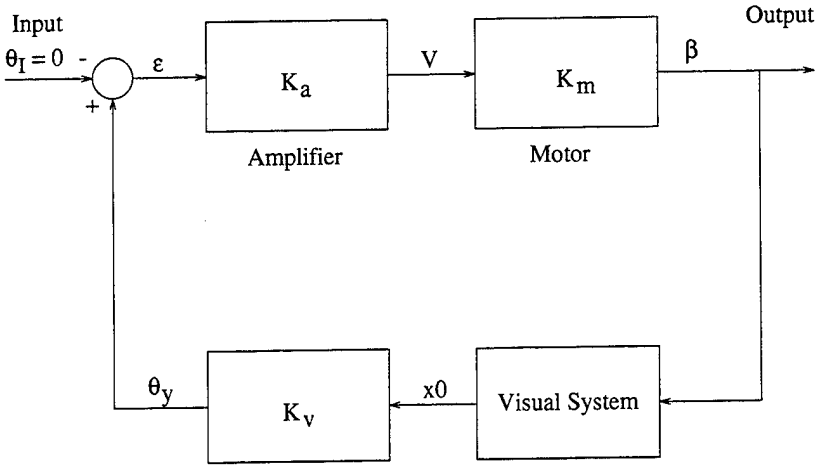


Figure 8: Servo control system.

We are now in a position to study various properties of the servo system. Referring to Figure 8, let the amplifier gain be K_a; then its voltage output V is given by

$$V = K_a\epsilon = K_a\theta_y \tag{15}$$

If the motor has no time lag and a speed at all times proportional to V,[2] then:

$$\beta = \frac{d\theta_y}{dt} = K_m V \tag{16}$$

where K_m represents the motor constant. Combining equations (15) and (16), we obtain the differential equation as

$$\frac{d\theta_y}{dt} = K\theta_y \tag{17}$$

where K is $K_a K_m$. The solution of (17) is given by $\theta = \theta_y^0 e^{-Kt}$, where θ_y^0 denotes the initial value of the pan.

3.4 Intermittent data

The servo considered thus far operates on the basis of error data supplied continuously. It is stable if the poles all lie in the half-plane to the left of the imaginary axis. However, in our actual system, the servo is actuated by error data supplied intermittently, at discrete moments equally spaced in time. The servo receives no information whatsoever about the error during the period between two consecutive pulses. We consider the case where the corrective action of the servo consists in continually exerting a torque on the output shaft in such a fashion that the torque is always proportional to the error found at the immediately preceding measurement. The torque then remains constant during the interval between two measurements, changing stepwise at each measurement. It is clear that overcorrection and instability will occur when the corrective torque per unit error is too large.

We can perform a standard analysis to convert a continuous system to a sampled-data one. Especially simple is the case when the continuous servo is controlled by (17). The transfer function $y(z)$ of the sampled-data version is given by

$$y(z) = \frac{K}{z + K}$$

In our present considerations, the unit circle plays the role that the half-plane to the left of the imaginary axis plays in the theory of continuous error input. Thus, the root of the denominator of $y(z)$ must be numerically less than 1. In other words, $|K|$ must be less than 1.

4 Task Two: Pursuing a Moving Target

The task discussed in the preceding section is limited to the case where the input is a step function, i.e., the desired heading direction is fixed. We next consider the case where the

[2]Actually, all motors have time constants, that is, exhibit inertial effects, and the differential equation must be modified by adding a term:

$$T_m \frac{d^2\theta_y}{dt^2} + \frac{d\theta_y}{dt} = K_m V$$

servo system is called upon to follow a time-varying input. We encounter such a situation when the system is involved in the pursuit of a moving target. In addition, we impose a second constraint on the control system: while tracking a target, we want the system to maintain a constant distance away from the target. Such a task requires more complex visual information processing. In particular, we compute from the images the time to collision, or time to contact [25], from the observer to the target, and we use it in the control of the robot.

We will now discuss the estimation of time to contact from images; we will also discuss related work and describe the technique used here. Let us note here that traditional methods by which this is accomplished cannot be used in our task. We first discuss the simple case of tracking a time-varying input via a PID controller. Then, an additional controller that makes use of the time to contact information is incorporated to meet our more elaborate criteria.

4.1 Estimating time to contact when the observer and the target are moving

The time to contact between the camera and a scene point is defined as the value $\frac{Z}{W}$, with Z being the depth and W the relative forward translational speed of the camera with respect to the point. If the scene point lies on the moving target, then assuming constant relative motion over time, $\frac{Z}{W}$ expresses the time left until the target will hit the infinitely large image plane. However, computing the time to contact of a point on the target is greatly complicated by the difficulty of computing the target's relative motion if the image of the target covers only a small part of the image plane. In the literature a number of methods have been proposed which are based on the utilization of divergence [10, 28, 32, 36]. However, divergence involves the computation of the spatial derivatives of flow and thus is very hard to compute. Also, divergence is proportional to two components, which cannot be separated without additional knowledge about the motion and the shape of the target. These are the time to contact and the product of the fronto-parallel translational velocity times a function of slope and tilt of the local plane in view. Another approach that has been proposed is based on the utilization of constraints from fixation and tracking [17] The change of rotation in order to track the target accurately is related to the target's change in depth which in turn is related to the time to contact. The problem with this approach is that it requires the robot to keep track of its own exact motion during tracking; this is a difficult task if the robot is considered not to be static, but moving itself.

The way to circumvent these problems is to use visual information not from the area covered by the target, but from the area surrounding the target. In particular, if the target is moving on the ground we can utilize the area at the bottom of the target, which usually is at about the same distance as the target. From this we can obtain a measure $\frac{Z}{W}$, where W is the forward translational velocity of the robot with respect to the static scene and Z is the depth of points close to the target.

Using the patterns described in Sections 3.1 and 3.2 we can estimate the motion of the observer. In particular, we can utilize the coaxis vectors parallel to the x-axis or the α-vectors very efficiently, even if there is additional rotational motion around the x-axis. In general, knowing the 3D motion and knowing the normal flow at a point allows us to derive

$\frac{Z}{W}$ at the corresponding point from (3) as

$$\frac{Z}{W} = \frac{(x - x_0)n_x + (y - y_0)n_y}{u_n - (u_{\text{rot}}n_x + v_{\text{rot}}n_y)} \tag{18}$$

For the copoint vectors with gradient $(1, 0)$ and the α-vectors, the value of the flow along the gradient can be approximated as

$$u_n = \frac{W}{Z}(x - x_0)$$

and thus

$$\frac{Z}{W} = \frac{x - x_0}{u_n}$$

Since we have estimated x_0 we can estimate the time to contact from this relationship by measuring the normal flow at a number of scene points and solving an overdetermined system of linear equations using a least square minimization.

4.2 Simple tracking

By simple tracking, we mean that the pursuer is not concerned with its distance from the target. Several additional considerations arise since the input is now time varying. Foremost among these is that the rise time and the damping should be improved. Furthermore, the type of the system should be increased so as to eliminate steady state error in response to a ramp input. Thus a PID controller is needed, where the PI portion is used to improve the steady-state error of the system, and the PD portion is used to meet the damping requirement. More generally, one can use a lead-lag controller to achieve the desired compensation.

Other techniques might be used to speed up the response of the system. Gain scheduling is one such method. Basically, gain scheduling approximates a nonlinear function with several linear pieces, each of which admits a linear control design. A typical example of such attempts is found in the design of an aircraft autopilot, where an airplane system model is linearized around several hundred operating points selected within the plane's operating region. In our case, the adjustable fixed gains are selected as a function of the FOE position. When the FOE is out of the image, a relatively high torque is applied to quickly bring the FOE within the field of view of the camera. As the system approaches the straight-ahead direction, the torque is "stepped down" according to a gain schedule. This improves the transient speed of response without bringing in steady-state oscillations.

There are various ways to improve the control. The most obvious is that, in order to track a maneuvering target well, we might use a more sophisticated tracking method for the camera that builds in some predictive capability. For instance, in Birmiwal et al. [7], two state models of the target were used: a low-order or nonmaneuvering model and a high-order or maneuvering model. When a maneuver was detected, a switch from the low-order to the high-order model was accomplished by adding extra state components. The tracking was then done with the augmented state model until reversion to the normal model took place as a result of another decision. The change in dimension of the filter allows acceleration to be modeled.

4.3 More elaborate pursuit

We next consider the case when the system not only has to pursue a moving target, but also has to keep a constant distance from it. To give an example from nature where such a task is required, consider the following situation: A male hoverfly *Syritta* shadows a potential mate until the latter lands on a flower. Only then does it attempt capture by accelerating with a constant force towards its target. The male while shadowing keeps a constant distance, about 10 cm, from its quarry [11]. Thus the strategy might consist of an initial phase during which the target is pursued with full speed. When the target is within a certain range, a switch in the control scheme is used to maintain the desired distance.

Such (and similar) problems in which additional constraints are imposed on the system typically lead to optimal solutions with highly complex switching control schemes, the solution of which is beyond the scope of our paper. Here we present a scheme that, given the information on time to contact, regulates its speed by a simple proportional controller.

Having estimated the time to contact as described before, we use this information to drive a separate servo loop that regulates the forward speed of the observer. To intercept the target as fast as possible, we set the desired time to contact to zero, and then use the difference between the desired and the computed time to contact to drive the servo loop. A desired time to contact with nonzero value will result in a kind of stalking behavior. Section 6, which describes our experiments, will present results on both these scenarios.

5 Task Three: Perimeter Following

5.1 Estimating functions of depth

More complex than the understanding of rigid motion is the understanding of depth and shape. Our viewpoint is that it is not necessary to compute exact depth measurements which are very hard to derive and whose computation requires exact knowledge about the motion (or stereo) configuration. Instead we could aim at computing less informative descriptions of shape and depth, such as functions of depth and shape where the functions are such that they can be computed easily from well-defined image information. These ideas are demonstrated here by means of the task of perimeter or wall following.

Perimeter following in our application is described as follows: A robot (car) is moving on a road which is bounded on one side by a wall-like perimeter. On the basis of visual information the robot has to control its steering in order to keep its distance from the perimeter at a constant value and maintain its forward direction as nearly parallel to the perimeter as possible. The perimeter is defined as a planar textured structure in the scene perpendicular to the plane of the road. This definition includes any planar structure (connected or not) that can be found at the boundary of a road or path, such as walls, houses parallel to the road, or a line of trees.

This task requires us to compute from the spatio-temporal images some form of information about the distance between the perimeter and the robot. The most common way to address this problem is to perform reconstruction in order obtain exact depth. Another approach requires us to compute the slopes of lines parallel to the road (boundary lines on

highways). This means that the boundary first has to be detected and thus the segmentation problem has to be solved. Unless the boundaries have image features, which are clearly distinguishable from the rest of the image, this is a very difficult task. In the approach described here we compute some form of "qualitative" depth information which is sufficient for deducing the steering motion and which we can derive without first having to compute the motion parameters and without having to detect boundary lines.

Our strategy applied to the perimeter following task is as follows. While the robot is moving forward it has its camera directed at some point on the perimeter. As it continues moving it maintains the relative orientation of the camera with regard to its forward translation. It compares distance information derived from flow fields obtained during its motion with distance information computed from a flow field obtained when it was moving parallel to the road. This distance information will tell what the robot's steering direction is with respect to the perimeter.

The distance information we use is the scaled directional derivative of inverse depth along (imaginary) lines on the perimeter. From the observed flow field, normal flow measurements along (imaginary) lines through the image center are selected and compared to normal flow measurements along (imaginary) lines of equal slope in the reference flow field. This information is derived from the image of the whole perimeter, and thus is global. Furthermore, as in the previous tasks, we do not need to compute correspondence or optical flow, but only the normal flow components.

5.2 Direct visual depth cue

As the robot is moving along its path the motion parameters perceived in the images change. For comparison reasons, we assume that the angle θ_y and the angle θ_x between the forward direction and the camera direction (determining x_0 and y_0) remain constant. The robot's rotational velocity (around the x-axis and y-axis) can change in any way. The technique, however, is independent of these parameters. We next investigate the motion fields perceived during motion and how depth is encoded in the flow values.

The motion perceived in the images is due to a translation (U, V, W) and a rotation (α, β). Thus from (3), if we divide u_n by n_x (if $n_x \neq 0$), we obtain a function $f_n(\mathbf{x}, \mathbf{n}) = \frac{u_n}{n_x}$ of the image coordinates $\mathbf{x} = (x, y)$ and the normal direction $\mathbf{n} = (n_x, n_y)$:

$$f_n(\mathbf{x}, \mathbf{n}) = \frac{u_n}{n_x} = \frac{(xW - U)}{Z} + \frac{(yW - V)}{Z}\frac{n_y}{n_x} + \alpha\left(\frac{xy}{f} + \left(\frac{y^2}{f} + f\right)\frac{n_y}{n_x}\right) - \beta\left(\left(\frac{x^2}{f} + f\right) + \frac{xy}{f}\frac{n_y}{n_x}\right). \quad (19)$$

Let us choose directions (n_x, n_y) such that $x + \frac{n_y}{n_x}y = K$ with K being some constant. These directions are the $(K, 0)$ copoint vectors, i.e. the vectors perpendicular to lines passing through the point $(K, 0)$. This can be easily seen by considering (n_x, n_y) as the tangents to a family of curves and solving the differential equation for $y(x)$:

$$\frac{n_y}{n_x} = y'(x) = \frac{K - x}{y(x)}$$

which has as its solution

$$y^2 + (x - K)^2 = C$$

For these directions (19) becomes

$$f_n(\mathbf{x},\mathbf{n}) = \frac{(-U - V\frac{n_y}{n_x} + KW)}{Z} + \alpha(f\frac{n_y}{n_x} + \frac{Ky}{f}) - \beta(f + \frac{Kx}{f}) \tag{20}$$

In the perimeter following task, we consider the normal flow along imaginary lines passing through the image center; thus $K = 0$ and we obtain

$$f_n(\mathbf{x},\mathbf{n}) = \frac{u_n}{n_x} = \frac{(-U - V\frac{n_y}{n_x})}{Z} + \alpha f\frac{n_y}{n_x} - \beta f \tag{21}$$

Along each of these lines $\frac{n_y}{n_x}$ is constant and thus $(-U - V\frac{n_y}{n_x})$ and $(\alpha f\frac{n_y}{n_x} - \beta f)$ are also constant and (21) describes $f_n(\mathbf{x},\mathbf{n})$ as a function which is linear in the inverse depth. For any two points P_1 and P_2 with coordinates $\mathbf{x}_1$ and $\mathbf{x}_2$ along such a line the difference $(f_n(\mathbf{x}_1,\mathbf{n}) - f_n(\mathbf{x}_2,\mathbf{n}))$ is independent of the rotation. We thus compute the directional derivative $D(f_n(\mathbf{x},\mathbf{n}))_{\mathbf{n}^\perp}$ of $f_n(\mathbf{x},\mathbf{n})$ at points on lines with slope $k = -\frac{n_x}{n_y}$ in the direction of a unit vector $\mathbf{n}^\perp = (-n_y, n_x)$ parallel to the image lines. Dropping, in the notation of the directional derivative, the dependence of f_n on $\mathbf{x}$ and $\mathbf{n}$ we obtain

$$D(f_n)_{\mathbf{n}^\perp} = (-U - V\frac{n_y}{n_x})D(\frac{1}{Z})_{\mathbf{n}^\perp} \tag{22}$$

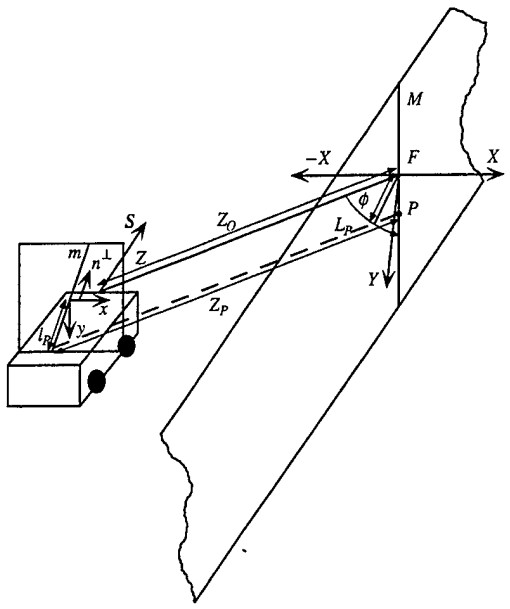


Figure 9: Geometric configuration during perimeter following.

We next derive $D(\frac{1}{Z})_{\mathbf{n}^\perp}$. Referring to Figure 9, the camera is mounted on the robot and the robot is moving forward with velocity S. The camera is directed toward a point F on the perimeter. We fix a coordinate system XYZ to the robot, such that the Z axis is aligned with the optical axis of the camera and the XY plane is perpendicular to it. Let m be the line in the image parallel to $\mathbf{n}^\perp$ along which we compute depth, and let M be the

corresponding line in 3D on the perimeter. Let Z_0 be the depth at the fixation point and Φ the angle between the Z-axis and M. The depth Z_P of any point P on M is

$$Z_p = Z_0 + \frac{L_p}{\tan \Phi}$$

where L_p is the value of the parallel projection of $\bar{FP}$ on the XY-plane. L_P projects perspectively onto l_P (along $\mathbf{n}^\perp$) in the image plane, where $l_P = \frac{L_P f}{Z_P}$; thus, dropping subscript P, for any depth value Z we obtain

$$Z = \frac{Z_0}{1 - \frac{l}{f \tan \Phi}}$$

For points with coordinates $l\mathbf{n}^\perp$ and gradient direction $\mathbf{n}$ we obtain

$$f_n(l\mathbf{n}^\perp, \mathbf{n}) = (-U - V\frac{n_y}{n_x})\frac{Z_0}{1 - \frac{l}{f \tan \Phi}} + \alpha(f\frac{n_y}{n_x} + \frac{Ky}{f}) - \beta(f + \frac{Kx}{f})$$

and (22) for any point along the line m becomes

$$D(f_n)_{\mathbf{n}^\perp} = (U + V\frac{n_y}{n_x})\frac{1}{\tan \Phi Z_0 f} \tag{23}$$

In the remainder of this section we demonstrate the dependence of $D(f_n)_{\mathbf{n}^\perp}$ on the robot's steering direction. In particular, we show that $|D(f_n)_{\mathbf{n}^\perp}|$ along certain directions $\mathbf{n}^\perp$ decreases as the robot steers towards the perimeter, or that for any two flow fields corresponding to motion configurations C_1 and C_2 depending on parameters (Φ_1, Z_{0_1}) and (Φ_2, Z_{0_2}), $|D(f_n)_{\mathbf{n}^\perp}|_1 > |(D(f_n)_{\mathbf{n}^\perp}|_2$ if $|\Phi_1| < |\Phi_2|$. Here, using the absolute value allows to provide a general notation that describes fixations of the robot to its left and to its right.

5.3 Comparing ordinal depth information

We describe vectors with respect to three orthogonal coordinate systems XYZ, $X'Y'Z'$ and $X''Y''Z''$ that are being rotated into each other (see Figure 10). The orientations of these coordinate systems are such that the Z-axis is parallel to the direction of translation when the robot is moving parallel to the perimeter, the Z'-axis is parallel to the robot's viewing direction in configuration C_1, and the Z''-axis is parallel to the robot's viewing direction in configuration C_2. The orientations of the frame $X'Y'Z'$ and the frame $X''Y''Z''$ are related to the orientation of the frame XYZ through rotation matrices R' and R'', as described below, which are dependent on parameters ϕ'_x, ϕ'_y and ϕ'_x, ϕ''_y (rotation around the x-axis is the same), where $|\phi''_y| > |\phi'_y|$.

$$R' = \begin{pmatrix} r'_{11} & r'_{12} & r'_{13} \\ r'_{21} & r'_{22} & r'_{23} \\ r'_{31} & r'_{32} & r'_{33} \end{pmatrix} = \begin{pmatrix} \cos \phi'_y & 0 & \sin \phi'_y \\ -\sin \phi'_y \sin \phi'_x & \cos \phi'_x & \cos \phi'_y \sin \phi'_x \\ -\sin \phi'_y \cos \phi'_x & -\sin \phi'_x & \cos \phi'_y \cos \phi'_x \end{pmatrix}$$

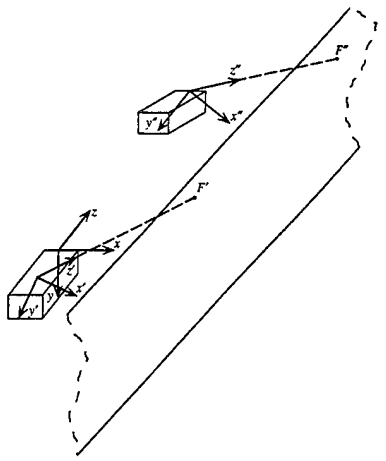


Figure 10: Three reference coordinate systems.

$$R'' = \begin{pmatrix} r''_{11} & r''_{12} & r''_{13} \\ r''_{21} & r''_{22} & r''_{23} \\ r''_{31} & r''_{32} & r''_{33} \end{pmatrix} = \begin{pmatrix} \cos\phi''_y & 0 & \sin\phi''_y \\ -\sin\phi''_y \sin\phi'_x & \cos\phi'_x & \cos\phi''_y \sin\phi'_x \\ -\sin\phi''_y \cos\phi'_x & -\sin\phi'_x & \cos\phi''_y \cos\phi'_x \end{pmatrix}$$

Any vector V in XYZ corresponds to V' in $X'Y'Z'$ and V'' in $X''Y''Z''$, where

$$V' = R'V \quad V'' = R''V \quad \text{and} \quad V = R'^T V' \quad V = R''^T V''$$

with $R'^T V'$ being the transpose of R' and R''^T being the transpose of R''.

Let (in C_1) m' be the line in the image with slope $k = -\frac{n_x}{n_y}$ along which measurements are taken, which is described by the following line equation:

$$m' : y' = kx'$$

In order to obtain a vector in 3D on the corresponding line M' on the perimeter, we intersect the plane $Y' = kX'$ with the plane of the perimeter, which is at a distance d from the robot and thus is described through equation $X = d$ in the coordinate system XYZ or $r'_{11}X' + r'_{21}Y' + r'_{31}Z' = d$ in the coordinate system $X'Y'Z'$. Thus a unit vector **a** along M' in $X'Y'Z'$ is computed as

$$\begin{pmatrix} 1 \\ k \\ \frac{-r'_{11}-kr'_{21}}{r'_{31}} \end{pmatrix} \frac{1}{\sqrt{1 + k^2 + \left(\frac{r'_{11}+kr'_{21}}{r'_{31}}\right)^2}}$$

and a unit vector **b** along the Z'-axis in $X'Y'Z'$ is

$$\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}.$$

The cosine of the angle Φ_1 between $\mathbf{a}$ and $\mathbf{b}$ is thus

$$\cos\Phi_1 = \mathbf{a}\cdot\mathbf{b} = -\frac{r'_{11}+kr'_{21}}{r'_{31}\sqrt{1+k^2+\left(\frac{r'_{11}+kr'_{21}}{r'_{31}}\right)^2}}$$

and

$$\tan\Phi_1 = -\frac{r'_{31}\sqrt{k^2+1}}{r'_{11}+kr'_{21}}$$

Let d be the distance from the camera to the perimeter. d is measured along the X-axis in XYZ. Any vector V in XYZ, which is parallel to the Z'-axis, is described as $\lambda(r'_{31}, r'_{32}, r'_{33})^T$, with λ being a scalar. Thus for a vector V of length Z_{O_1}, $d = \lambda r'_{31}$ and the value of Z_{O_1} amounts to

$$Z_{O_1} = \frac{d}{r'_{31}}$$

and thus

$$\frac{1}{\tan\Phi_1 Z_{O_1}} = -\frac{r'_{11}+kr'_{21}}{d\sqrt{k^2+1}} \tag{24}$$

Assuming that the time between measurements is small, and thus the horizontal distance d in C_2 is the same as in C_1, we obtain for C_2 (see Figure 11)

$$\frac{1}{\tan\Phi_2 Z_{O_2}} = -\frac{r''_{11}+kr''_{21}}{d\sqrt{k^2+1}} \tag{25}$$

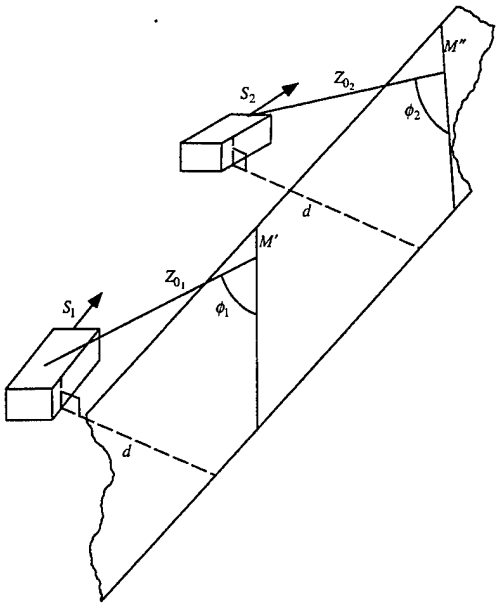


Figure 11: Comparing the values from two configurations.

Comparing (24) and (25), we are thus interested in values k for which

$$|r'_{11}+kr'_{21}| > |r''_{11}+kr''_{21}| \tag{26}$$

or

$$|\cos\phi'_y - k\sin\phi'_x\sin\phi'_y| > |\cos\phi''_y - k\sin\phi'_x\sin\phi''_y|$$

This inequality is true for all ($k\sin\phi'_x\sin\phi'_y > 0$), assuming that $|\phi'_y| < 45°$ and $|\phi''_y| < 45°$. (This assumption is used only to allow for a general notation and is not needed if different sign cases are listed separately.) On the basis of this result the input to the control system for deriving the steering direction can be generated. At any point in time, $D(f_n)_{\mathbf{n}^\perp}$ for directions $\mathbf{n}^\perp$ such that ($-\frac{n_x}{n_y}\sin\phi'_x\sin\phi'_y > 0$) are computed from the visual flow field and compared to prestored values $D(f_n)_{\mathbf{n}^\perp}$ due to a forward motion parallel to the perimeter; the computed sign of the change in $|D(f_n)_{\mathbf{n}^\perp}|$ defines the sign of the change in the steering angle.

In the analysis above, the assumption was made that the horizontal distance d does not change. When this assumption does not hold, the same principle can still be applied to a smaller number of values if enough image gradients are available. If the distance d decreased between C_1 and C_2, then $|r'_{11} + kr'_{21}|$ may be smaller than $|r''_{11} + kr''_{21}|$ for values of $|k|$ smaller than some threshold T, but must be larger for values of $|k|$ greater T, and thus if gradients on a line with $|k| > T$ are available, these measurements can still be used for comparison.

Finally, we want to characterize the lines for which ($k\sin\phi'_x\sin\phi'_y > 0$). Comparing such a line to the image of a line parallel to the road on the perimeter passing through the image center, we find that the slopes of the two lines are of opposite sign. For example, if the camera's optical axis is pointing down and to the right (as in Figure 9) then ϕ'_x is negative and ϕ'_y is positive and the slope of the parallel line is positive, whereas the slope of the line we use for comparison has to be negative.

6 Experiments

This section presents the results of simulations and results obtained with real images using the algorithms proposed in the previous sections.

A. Task 1

The first experiments used simulations of the robot's trajectory. The robot is initially moving at an angle relative to the z-axis. It is then required to steer itself so that it is heading along the z-axis, where the feature $\mathbf{p}$ is located at a distance of 5 m away. The robot is moving with a constant forward speed of 1.5 m/s.

The mobile platform is a conventionally steered vehicle; the instantaneous radius of curvature r of its trajectory is related to the steering angle of its wheel θ_R as follows:

$$r = \frac{0.5L}{\sin\theta_R}$$

where L is the body length of the vehicle. The field of view of the camera used in the simulation is 50°. We created the synthetic normal flow field from a scene with random depth and we added zero-mean Gaussian noise with a standard deviation of 1 pixel to the normal flow measurements. For the estimation of x_0, we employed the α-vectors, where (as

described in Section 3.2) we approximated the α-hyperbola by a straight line, which was estimated using a linear classifier.

Figure 12 shows the trajectories for two different values of the proportionality constant K generated by the servo system. The two curves correspond to values of $K = 0.1$ and $K = 0.3$. As can be seen, the system has a poor rise time and insufficient damping—typical of a proportional controller.

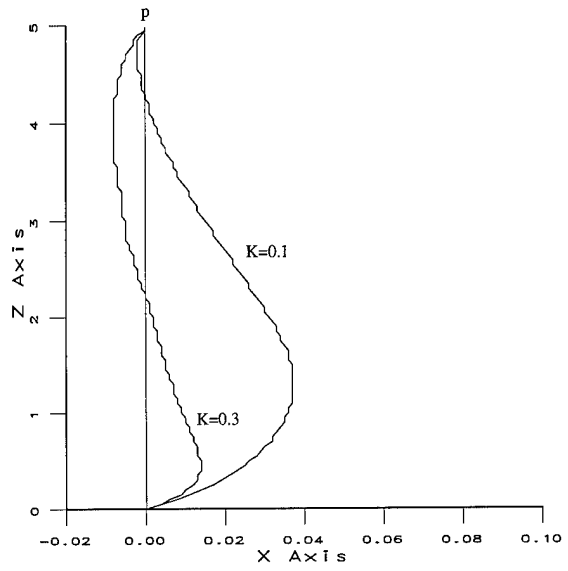


Figure 12: Trajectory generated by the servo system.

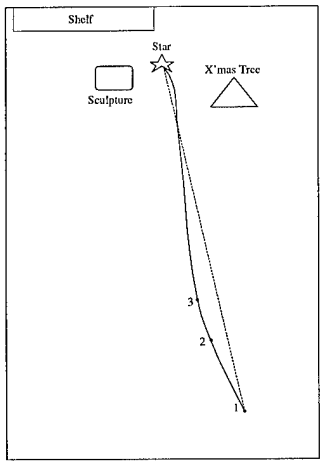


Figure 13: 3D configuration as studied in Task 1.

In our experiments with real images, the mobile platform on which the camera was mounted was a conventionally steered vehicle. The camera had a focal length of 1136 pixels, and the image dimensions were at 720×576; thus the field of view was approximately 30°. Considering that the steering movements must be small so as to reduce inter-frame

disparity, and allowing for the computation of image flow, especially since the focal length of the camera was very large, we chose to operate the servo system with a proportionality constant of $K = 0.1$. Figure 14 shows some images taken by a camera mounted on a mobile platform, as the latter is making steering movements. The feature $\mathbf{p}$ corresponds to the star in the center of the image, mounted on a tripod and initially located at a distance of 5 m from the camera. Figure 13 displays the configuration of this setting including the robot's trajectory. Figure 14a, c, and e show images taken by the system at three time instants (as marked in Figure 13) with the normal flow fields superimposed. Figures 14b, d, and f show the positive and negative α-vectors as computed from the normal flow fields in black and white and the line approximating the α-hyperbola which has been fitted to the data.

Figure 14: Task 1: Some scenes along the trajectory.

B. Task 2

The experiments on Task 2 were based on simulations. In the simple type of pursuit, the servo operates based on the misalignment between the visual axis and the axis of the robot body. It is not concerned with the distance between the robot and the target. In other words, the forward speed of the robot is not regulated.

Figure 15 depicts the trajectories computed for the following two scenarios. In Figure 15a, the target was moving to the left with a speed of 0.5 m/s; there was no motion along the z-axis. In Figure 15b, the target was receding in depth as well. The target was initially located 10 m away from the robot. In both cases the servo was operated based on a PID controller, with $K_P = 0.3, K_I = 0.03$, and $K_D = 0.1$, where K_P, K_I, and K_D are the proportionality constant, the integral constant, and the derivative constant, respectively. The trajectory of the target is shown as a solid line, and that of the robot is shown as a dotted curve. As can be seen, compared to Figure 12, the system now has much better transient characteristics, due to its PID control.

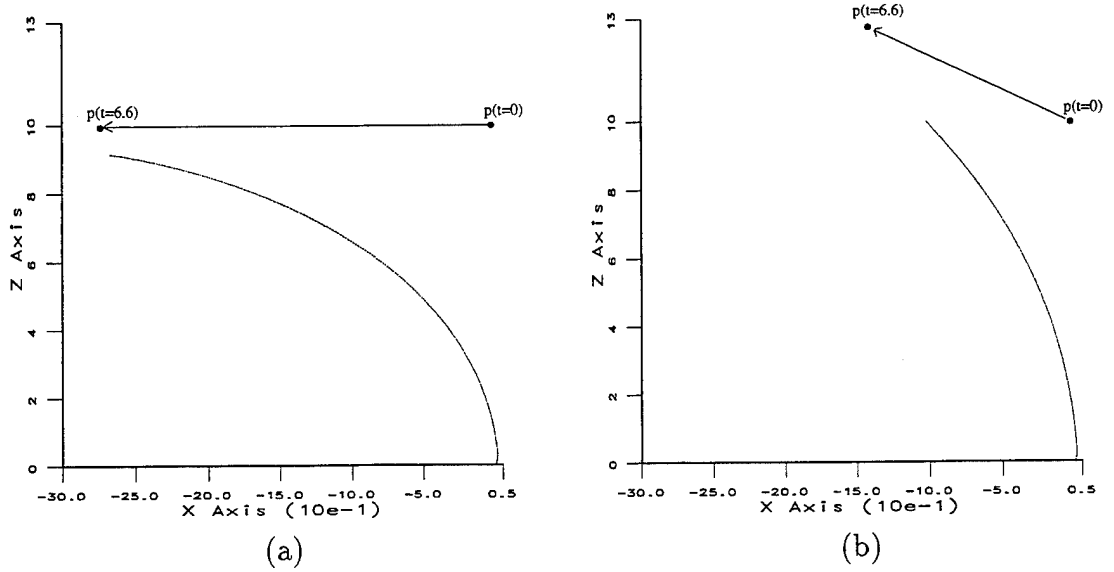


Figure 15: Simple pursuit of a moving target: (a) Target moving along the x-axis. (b) Target moving off at an angle. The dotted path represents the trajectory of the robot; the solid line represents that of the target.

For the purpose of target interception, in addition to the above control, forward speed regulation is required. Figure 16a shows the case where the speed is regulated so as to minimize the distance between the target and the pursuer as much as possible, subject to the speed limit of the robot. It can be seen that the target was captured within a much shorter time. Figure 16b shows the case where the criterion is to maintain a constant time to contact. As described in Section 4.3 the control was operated by setting the desired time to contact to zero and using the difference between the desired and the computed time to contact to drive the servo loop. The study shows that as the robot approaches the target, it

slows down accordingly. Thus a kind of shadowing behavior is exhibited, and the interception of the target can be effected in a much smoother manner.

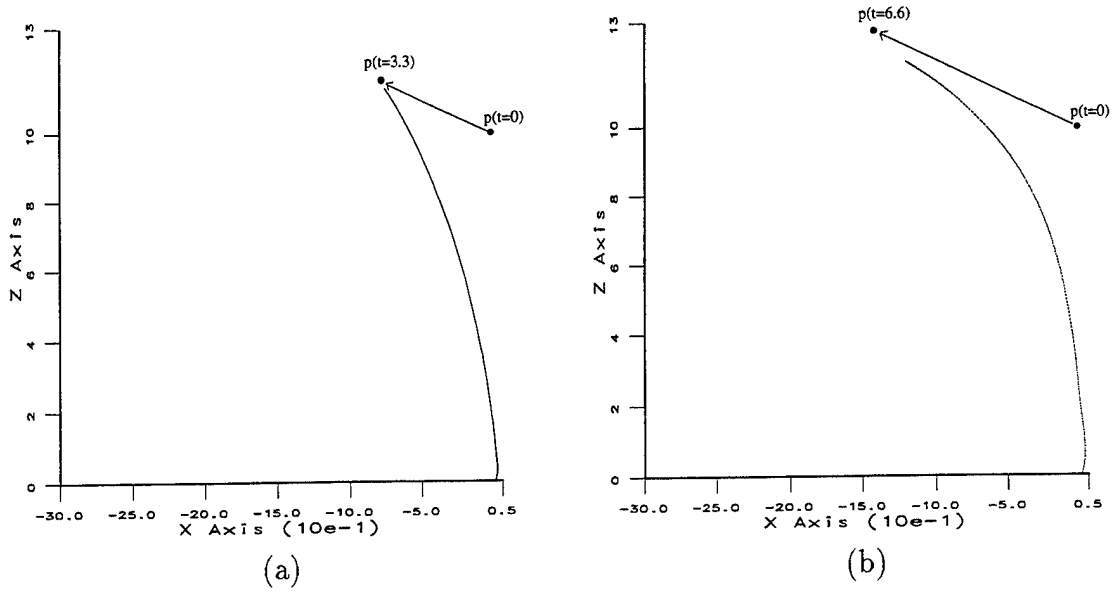


Figure 16: More elaborate pursuit of a moving target: (a) To minimize time to contact. (b) To maintain a constant time to contact. The dotted path represents the trajectory of the robot; the solid line represents that of the target.

C. Task 3

In the experiments on Task 3 a mobile platform with a camera mounted on it moved along an alley-like perimeter. The camera had a focal length of about 1000 pixels and the image dimensions were 512×512. The servomechanism was implemented as a simple proportional control relating the robot's rotational speed around the y-axis to the directional derivative of f_n (as defined in Section 5.2). The scene contained a highly textured perimeter. We thus derived image measurements along a number of lines and used the mean of the computed estimates as input to the servo system. Figure 17a shows one of the reference images, which was taken when the robot was moving parallel to the perimeter. The lines along which image measurements were taken are overlaid on the image in white. Figure 17b shows the normal flow field computed for this same image. Figure 17c and d display two more images taken while the robot moved along its path, one when it steered towards the perimeter and another one later when it again moved away from the perimeter.

7 Conclusions

A new way of making use of visual information for autonomous behavior has been presented. Visual representations which are manifested through geometrical constraints defined on the

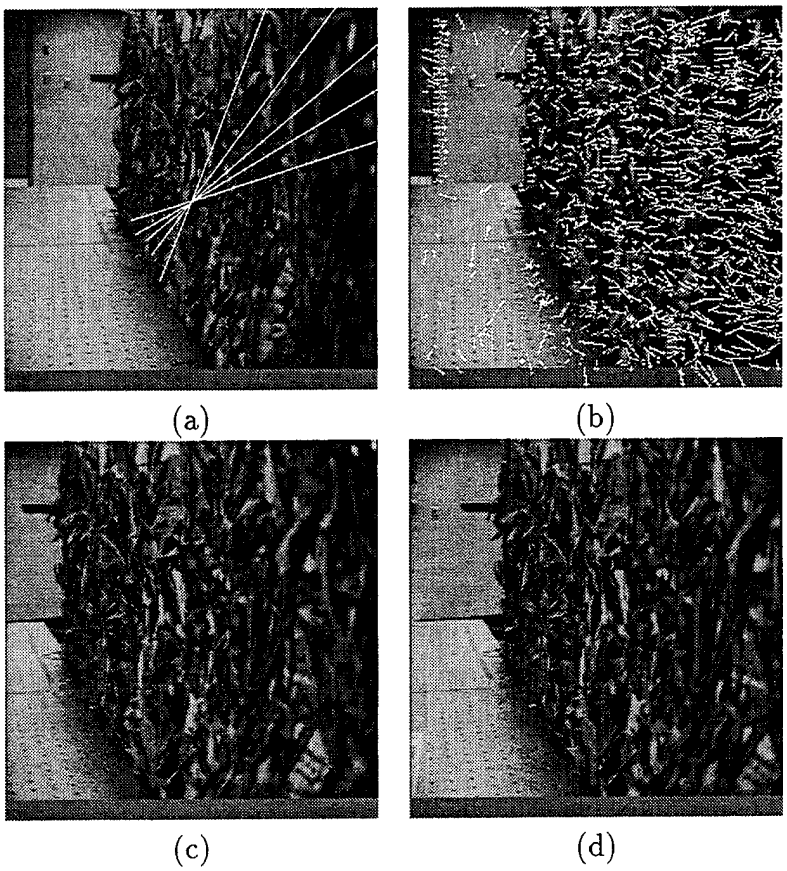


Figure 17: Task 3: The robot moves along an "alley".

flow in various directions and on the normal flow were used as input to the servomechanism. Specifically, the constraints described are global patterns in the sign of the flow in different directions whose positions and forms are related to 3D motion, and patterns of normal flow along lines in the image which encode relative 3D distance information. 3D motion and structure representations derived from these constraints were applied to the solution of a number of navigational problems involving the control of a system's 3D motion with respect to its environment and to other moving objects. Some of the constraints described, however, are of a general nature, and thus might be utilized in various modified forms for the solution of other navigational problems.

References

[1] J. (Y.) Aloimonos, Purposive and qualitative active vision. In *Proc. DARPA Image Understanding Workshop*, pages 816–828, 1990.

[2] J. Aloimonos, I. Weiss, and A. Bandopadhay, Active vision. *International Journal of Computer Vision*, 2:333–356, 1988.

[3] P. Anandan and R. Weiss, Introducing a smoothness constraint in a matching approach for the computation of optical flow fields. In *Proc. 3rd Workshop on Computer Vision: Representation and Control*, pages 186–194, 1985.

[4] R.C. Arkin, R. Murphy, M. Pearson and D. Vaughn, Mobile robot docking operations in a manufacturing environment: Progress in visual perceptual strategies. In *Proc. IEEE International Workshop on Intelligent Robots and Systems*, pages 147–154, 1989.

[5] R. Bajcsy, Active perception. *Proc. of the IEEE*, 76:996–1005, 1988.

[6] D. Ballard and C. Brown, Principles of animate vision. *CVGIP: Image Understanding*, 45:3–21, Special Issue on Purposive, Qualitative, Active Vision, Y. Aloimonos (Ed.), 1992.

[7] K. Birmiwal and Y. Bar-Shalom, On tracking a maneuvering target in clutter. *IEEE Trans. on Aerospace and Electronic Systems*, 20:635–644, 1984.

[8] R.A. Brooks A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2:14–23, 1986.

[9] A. Bruss and B. Horn, Passive navigation. *Computer Vision, Graphics, and Image Processing*, 21:3–20, 1983.

[10] R. Cipolla and A. Blake, Surface orientation and time to contact from image divergence and deformation. In *Proc. European Conference on Computer Vision,* pages 187–202, 1992.

[11] T.S. Collett and M.F. Land, Visual control of flight behavior in the hoverfly *Syritta pipiens. J. Comp. Physiol.*, 99:1–66, 1975.

[12] R. Duda and P. Hart, *Pattern Classification and Scene Analysis.* Wiley, New York, 1962.

[13] B. Espiau, F. Chaumette, and P. Rives, A new approach to visual servoing in robotics. *IEEE Trans. on Robotics and Automation*, 8:313–326, 1992.

[14] E. Francois and P. Bouthemy, Derivation of qualitative information in motion analysis. *Image and Vision Computing,* 8:279–288, 1990.

[15] O. Faugeras, *Three Dimensional Computer Vision.* MIT Press, Cambridge, MA, 1992.

[16] C. Fermüller, Navigational preliminaries. In Y. Aloimonos (Ed.), *Active Perception*, Advances in Computer Vision, pages 103–150. Lawrence Erlbaum, Hillsdale, NJ, 1993.

[17] C. Fermüller and Y. Aloimonos, Tracking facilitates 3-D motion estimation. *Biological Cybernetics*, 67:147–158, 1992.

[18] C. Fermüller and Y. Aloimonos, Qualitative egomotion. *International Journal of Computer Vision*, 15:7–29, 1995.

[19] C. Fermüller and Y. Aloimonos, On the geometry of visual correspondence. *International Journal of Computer Vision*, to appear, 1995.

[20] J-Y. Hervé, R. Sharma and P. Cucka, Toward robust vision-based control: Hand/eye coordination without calibration. In *Proc. IEEE International Symposium on Intelligent Control*, pages 147–154, 1991.

[21] E. Hildreth, *The Measurement of Visual Motion.* MIT Press, Cambridge, MA, 1983.

[22] B. Horn, Relative orientation. *International Journal of Computer Vision*, 4:59–78, 1990.

[23] B. Horn and B. Schunck, Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[24] J. Koenderink and A. van Doorn, Local structure of movement parallax of the plane. *Journal of the Optical Society of America*, 66:717–723, 1976.

[25] D.N. Lee, The optic flow field: The foundation of vision. *Phil. Trans. Royal Society London*, 290, 1980.

[26] H. Longuet-Higgins, A computer algorithm for reconstruction of a scene from two projections. *Nature*, 293:133–135, 1981.

[27] H.C. Longuet-Higgins and K. Prazdny, The interpretation of a moving retinal image. *Proc. Royal Society London B*, 208:385–397, 1980.

[28] R.C. Nelson and Y. Aloimonos, Obstacle avoidance using flow field divergence. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11:1102–1106, 1989.

[29] K. Pahlavan and J.-O. Eklundh, A head-eye system—Analysis and design. *CVGIP: Image Understanding*, 56:41–56, Special Issue on Purposive, Qualitative, Active Vision, Y. Aloimonos (Ed.), 1992.

[30] D. Raviv and M. Herman, A new approach to vision and control for road following. In *Proc. IEEE Workshop on Visual Motion*, pages 217–225, 1991.

[31] J. Santos-Victor, G. Sandini, F. Curotto and S. Garibaldi, Divergent stereo for robot navigation: Learning from bees. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 434–439, 1993.

[32] G. Sandini, F. Gandolfo, E. Grosso and M. Tistarelli, Vision during action. In Y. Aloimonos (Ed.), *Active Perception*, Advances in Computer Vision, pages 151–190. Lawrence Erlbaum, Hillsdale, NJ, 1993.

[33] A. Singh, *Optic Flow Computation: A Unified Perspective.* PhD thesis, Department of Computer Science, Columbia University, New York, NY, 1990.

[34] S.B. Skaar, W.H. Brockman, and R. Hanson, Camera-space manipulation. *International Journal of Robotics Research*, 6:20–32, 1987.

[35] M. Spetsakis and J. Aloimonos, Optimal computing of structure from motion using point correspondence. In *Proc. International Conference on Computer Vision*, pages 449–453, 1988.

[36] M. Subbarao, Bounds on time-to-collision and rotational component from first-order derivatives of image flow. *Computer Vision, Graphics, and Image Processing*, 50:329–341, 1990.

[37] R. Tsai and T. Huang, Uniqueness and estimation of three–dimensional motion parameters of rigid objects with curved surfaces. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 6:13–27, 1984.

[38] S. Ullman, *The Interpretation of Visual Motion.* MIT Press, Cambridge, MA, 1979.

[39] A. Verri and T. Poggio, Against quantitative optic flow. In *Proc. International Conference on Computer Vision*, 171–180, 1987.

[40] L.E Weiss and A.C. Sanderson, Dynamic sensor-based control of robots with visual feedback. *IEEE Trans. on Robotics and Automation*, 3:404–417, 1987.

[41] R.S. Woodworth, The accuracy of voluntary movement. *Psychological Review: Series of Monograph Supplements*, Suppl. 3, no. 3, July 1899.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	July 1995	Technical Report

4. TITLE AND SUBTITLE
3D Motion and Shape Representations in Visual Servo Control

5. FUNDING NUMBERS
IRI-90-57934
N00014-93-1-0257

6. AUTHOR(S)
Cornelia Fermüller, LoongFah Cheong, and Yiannis Aloimonos

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Computer Vision Laboratory
Center for Automation Research
University of Maryland
College Park, MD 20742-3275

8. PERFORMING ORGANIZATION REPORT NUMBER
CAR-TR-779
CS-TR-3499

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Office of Naval Research
Atlanta Regional Office
101 Merietta Tower, Suite 2805
Atlanta, GA 30323-0008

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release.
Distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The study of visual navigation problems requires the integration of visual processes with motor control processes. Most essential in approaching this integration is the study of appropriate spatio-temporal representations which the system computes from the imagery and which serve as interfaces to all cognitive and motor activities. Since representations resulting from exact quantitative reconstruction have turned out to be very hard to obtain, we argue here for the necessity of of representations which can be computed easily, reliably and in real time and which recover only the information about the 3D world which is really needed in order to solve the navigational problems at hand. In this paper we introduce a number of such representations capturing aspects of 3D motion and scene structure which are used for the solution of navigational problems implemented in visual servo systems.

14. SUBJECT TERMS
Visual servo control, structure from motion, qualitative reconstruction

15. NUMBER OF PAGES
36

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

GENERAL INSTRUCTIONS FOR COMPLETING SF 298

The Report Documentation Page (RDP) is used in announcing and cataloging reports. It is important that this information be consistent with the rest of the report, particularly the cover and title page. Instructions for filling in each block of the form follow. It is important to ***stay within the lines*** to meet ***optical scanning requirements.***

Block 1. Agency Use Only *(Leave blank)*.

Block 2. Report Date. Full publication date including day, month, and year, if available (e.g. 1 Jan 88). Must cite at least the year.

Block 3. Type of Report and Dates Covered. State whether report is interim, final, etc. If applicable, enter inclusive report dates (e.g. 10 Jun 87 - 30 Jun 88).

Block 4. Title and Subtitle. A title is taken from the part of the report that provides the most meaningful and complete information. When a report is prepared in more than one volume, repeat the primary title, add volume number, and include subtitle for the specific volume. On classified documents enter the title classification in parentheses.

Block 5. Funding Numbers. To include contract and grant numbers; may include program element number(s), project number(s), task number(s), and work unit number(s). Use the following labels:

C	- Contract	**PR**	- Project
G	- Grant	**TA**	- Task
PE	- Program Element	**WU**	- Work Unit Accession No.

Block 6. Author(s). Name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. If editor or compiler, this should follow the name(s).

Block 7. Performing Organization Name(s) and Address(es). Self-explanatory.

Block 8. Performing Organization Report Number. Enter the unique alphanumeric report number(s) assigned by the organization performing the report.

Block 9. Sponsoring/Monitoring Agency Name(s) and Address(es). Self-explanatory.

Block 10. Sponsoring/Monitoring Agency Report Number. *(If known)*

Block 11. Supplementary Notes. Enter information not included elsewhere such as: Prepared in cooperation with...; Trans. of...; To be published in.... When a report is revised, include a statement whether the new report supersedes or supplements the older report.

Block 12a. Distribution/Availability Statement. Denotes public availability or limitations. Cite any availability to the public. Enter additional limitations or special markings in all capitals (e.g. NOFORN, REL, ITAR).

- **DOD** - See DoDD 5230.24, "Distribution Statements on Technical Documents."
- **DOE** - See authorities.
- **NASA** - See Handbook NHB 2200.2.
- **NTIS** - Leave blank.

Block 12b. Distribution Code.

- **DOD** - Leave blank.
- **DOE** - Enter DOE distribution categories from the Standard Distribution for Unclassified Scientific and Technical Reports.
- **NASA** - Leave blank.
- **NTIS** - Leave blank.

Block 13. Abstract. Include a brief *(Maximum 200 words)* factual summary of the most significant information contained in the report.

Block 14. Subject Terms. Keywords or phrases identifying major subjects in the report.

Block 15. Number of Pages. Enter the total number of pages.

Block 16. Price Code. Enter appropriate price code *(NTIS only)*.

Blocks 17. - 19. Security Classifications. Self-explanatory. Enter U.S. Security Classification in accordance with U.S. Security Regulations (i.e., UNCLASSIFIED). If form contains classified information, stamp classification on the top and bottom of the page.

Block 20. Limitation of Abstract. This block must be completed to assign a limitation to the abstract. Enter either UL (unlimited) or SAR (same as report). An entry in this block is necessary if the abstract is to be limited. If blank, the abstract is assumed to be unlimited.